**Z7 – Rad na završnom projektu MMS iz ranijih godina**

Projekt koji se obrađivao: Igrica – kviz (autorica Ana Mimica), 2015./'16. god.  
Projekt dorađivala i analizirala Ivona Raguž.

U nastavku teksta bit će opisana uočena moguća poboljšanja koda izabranog projekta, proširenje funkcionalnosti te uočavanje pojedinih problema koji bi mogli biti predmetom daljnje analize i unaprijeđenja projekta.  
Rad na ovom projektu prvenstveno je bio baziran na poboljšanju koda te poboljšanju njegove čitljivosti. Dodana je i osnovna dokumentacija koda koja u originalnom projektu nije postojala.

Sve promjene i dorade koje će se opisati u nastavku vidljive su na sljedećem linku: <https://github.com/ivona13/MMS2019>.

Opis uočenih problema vezanih u samu preglednosti koda:

* Polje stringova *poljePitanja* čuva informacije o pitanjima, potencijalnim odgovorima te točnom odgovoru. Za daljnju nadogradnju projekta smatram da bi ovakve informacije o pitanjima trebalo odvojiti od same implementacije. U mojoj nadogradnji projekta, pitanja i odgovore odvojila sam u poseban file pod nazivom *data.csv*. U tablicu učitamo podatke iz *data.csv*, s *options* parametrom jednakim *header*. Vrlo jednostavno dobavljamo podatke (pitanja, odgovore, točan odgovor) iz tablice.
* Klasa *Pitanje* sadrži možda i previše informacija, a sadržavala bi još i više kada bi, recimo, umjesto četiri moguća odgovora htjeli pet. Također se pamti pozicija svakog odgovora, što bi se vrlo jednostavno moglo riješiti fiksiranjem razmaka između svih odgovora, pa bi tako pamtili samo informaciju o položaju pitanja. Pristup za koji sam se odlučila u ovoj analizi koda je korištenje dviju mapa. Prva mapa za ključ ima pitanje, a vrijednost tog ključa je polje stringova koje predstavlja polje odgovora. Druga mapa za ključ ima pitanje, a vrijednost ključa je redni broj točnog odgovora (kao string). Definirala sam funkciju *question(Map<String, ArrayList<String> >, int)* koja dohvaća pitanje pod zadanim rednim brojem. Ova funkcija mi treba pri generiranju popisa pitanja koji će se prikazivati u kvizu. Slijedim primjer iz originalnog koda o broju pitanja kviza – 10. Broj pitanja se može mijenjati kroz varijablu *numberOfQuestions*. U izvornom kodu nema varijable koja bi čuvala informaciju o broju pitanja, pa nije izbjegnuta *magic number* pojava. Pitanja se izabiru slučajnim odabirom – generiranjem cijelog broja funkcijom *random*, a potom uzimanjem pitanja pod tim istim rednim brojem.
* Funkcija *check(char)* u klasi *Pitanje* sadrži previše *if*-ova koji su ekvivalentni, što bi za veći broj mogućih odgovora postalo više skoro istih linija koda. Vrlo lako bi se mogla napraviti usporedba tocnog odgovora (u izvornom kodu tipa *int*) te vrijednosti *keyToTest – '0'*, što je također tipa *int*. Time bi se izbjeglo suvišno dupliciranje koda.
* U kodu postoji nekih varijabli za koje nije baš jasno što znače – nedostaje malo sugestivnosti (naprimjer, varijabla *h*).

Novi elementi koji su uvedeni:

* Na prijedlog autora izvornog koda, dodana je ograničenost trajanja vremena koje korisnik ima na raspolaganju za odgovaranje na pitanje. Ta ograničenost iznosi 4 sekunde. U jednom dijelu ekrana vidljivo je odbrojavanje sekundi preostalih za pokušaj odgovora. U slučaju da korisnik za 4 sekunde nije odgovorio na pitanje, na ekranu će se pojaviti sljedeće pitanje.
* Kada korisnik ponudi odgovor na pitanje, neće se dogoditi nestajanje pitanja iz prozora, nego će korisnik dobiti povratnu informaciju o točnosti odgovora nekom od poruka *Točan odgovor* ili *Vaš odgovor je netočan. Ovo je točan odgovor.* Ono što je bitno napomenuti o poziciji ove povratne informacije je sljedeće: u slučaju da je korisnik točno odgovorio na pitanje, potvrdna poruka nalazit će se u retku točnog (tj. ponuđenog) odgovora. U slučaju da korisnik odgovori pogrešno, poruka *Vaš odgovor je netočan. Ovo je točan odgovor.* ispisat će se u retku onog odgovora koji je točan. Razlog zašto je korigirana originalna ideja o povratnoj informaciji je sljedeći:
  + u slučaju da korisnik pogrešno odgovori na pitanje, ne dobije povratnu informaciju o tome koji odgovor je točan,
  + u slučaju da je korisnik odgovorio na pitanje točno, može se dogoditi da je točan odgovor sasvim slučajno pogođen pa bi bilo lijepo imati uvid i u ostale odgovore.
* Korisniku je u svakom trenutku kviza vidljivo koliko pitanja je do sada postavljeno i na koliko njih je točno odgovorio. Po završetku kviza, korisnik dobije uvid u postotak točnosti odgovora na pitanja.
* Na ekranu je ispisano i vrijeme koje je korisnik proveo igrajući kviz.

Dodatne mogućnosti koje bi se mogle implementirati u daljnjoj razradi projekta:

* Moglo bi postojati više razina kviza – primjerice, razlika među pojedinim razinama mogla bi, osim težine pitanja, biti i dopušteno vrijeme nuđenja odgovora (level 1 – 10 sekundi, level 2 – 8 sekundi, ...).
* Mogla bi se uvesti područja s pitanjima – postojao bi izbornik u kojem je moguće birati iz kojeg područja želimo pitanja.
* Bilo bi moguće pojedino područje označiti kao omiljeno te se natjecati s drugim korisnicima.
* Kao implementacijski detalj, bilo bi moguće primjerice ne ograničiti broj odgovora na 4 (u *.csv* file-u ne odrediti broj stupaca s *odg1*, *odg2, odg3*, *odg4*), nego definirati jedan stupac *odgovor* te osmisliti način međusobnog odvajanja odgovora, pa funkcijom *split* međusobno ih razdvojiti pri dohvaćanju.